Competitive behaviour of deep learning algorithms

Jens T. Grüb*

September 4, 2023

Abstract

Studies have shown that tacit collusion can occur in experiments with reinforcement learning algorithms, namely tabular Q-learning. Typically, the algorithm types and parameter settings are identical in such experiments. In this study, heterogeneous algorithms and parameter settings are used. Additionally, deep reinforcement learning algorithms are used instead of tabular Q-learning. Experiments with heterogeneity show significant differences in the average profit gains compared to homogeneous experiments. In accordance to existing studies, the implementation phase of reinforcement learning algorithms is seen as critical to the emergence of supracompetitive prices.

JEL-Codes: D43, D83, L13, L49
Keywords: pricing algorithms, algorithmic collusion, deep learning

Abstract

Studies have shown that tacit collusion can occur in experiments with reinforcement learning algorithms, namely tabular Q-learning. Typically, the algorithm types and parameter settings are identical in such experiments. In this study, heterogeneous algorithms and parameter settings are used. Additionally, deep reinforcement learning algorithms are used instead of tabular Q-learning. Experiments with heterogeneity show significant differences in the average profit gains compared to homogeneous experiments. In accordance with existing studies, the implementation phase of reinforcement learning algorithms is seen as critical to the emergence of supra-competitive prices.

*j.grueb@uni-hohenheim.de. Universität Hohenheim, Institut für Volkswirtschaftslehre, Lehrstuhl für Mikroökonomik (520C), 70593 Stuttgart, Germany. For helpful comments I want to thank Ulrich Schwalbe, Nadja Dwenger, Franziska Schünemann, Matthias Muijs, Katrin Buchali and the participants of the Hohenheim Brown Bag Seminar 2022. All remaining errors are mine. The author has no conflict of interest to declare. The author received no financial support for this article.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

1 Introduction

Algorithmic collusion is discussed extensively among scholars of economics, law and computer sciences as machine learning technologies evolve quickly and are increasingly deployed in businesses. Such algorithms are able to learn how to set reasonable prices. The main concern is that machine learning algorithms not just set competitive prices but learn to cooperate and set collusive prices without being explicitly programmed to do so (Ezrachi and Stucke, 2017; Harrington, 2018; Gal, 2019). On the other side, it is argued that collusion between two or more learning algorithms is not likely to emerge because there are still technical limitations (Schwalbe, 2018; Dorner, 2020).

There are indeed experiments showing that collusion emerges in experiments with machine learning algorithms. Calvano et al. (2020) show that two instances of a tabular Q-learning algorithm can learn to charge supra-competitive prices in an economic environment with logit demand where both agents interact simultaneously and repeatedly. Similarly, Klein (2021) also uses tabular Q-learning with two agents to show that collusive behavior can be learned by algorithms. The economic background of the environment in his study is the infinitely repeated sequential pricing duopoly of Maskin and Tirole (1988).

Experiments are not limited to tabular Q-learning. In some studies, more advanced deep reinforcement learning methods, such as Deep Q-Networks (DQN), are used in economic environments. Hettich (2021) compares tabular Q-learning to DQN. He is able to show that, compared to tabular Q-learning, DQNs can learn to collude significantly faster in the mentioned logit demand environment with two agents. Running a tabular Q-learning agent against a DQN agent leads to an asymmetric result in which the DQN agent receives profits equal to half the monopoly profit while the tabular Q-learning agent receives Nash profits. Additionally, he adds more firms to the experiments such that up to ten firms are competing against each other which leads to very low or no collusion at all.

In addition to that Eschenbaum, Mellgren, and Zahn (2022) investigate whether trained tabular Q-learning algorithms are able to transfer the learned behavior to new environments. They first show that collusion breaks down when transferring to a new environment. The authors continue with a restriction of the state space of agents such that only the own price of an agent in the last period defines the state instead of the prices of all agents in the last state. This way the authors are able to show that restricting the state space makes collusion robust when transferring to new environments.

The present study adds to this literature by asking the question of how algorithmic heterogeneity influences the level of algorithmic collusion. Algorithmic heterogeneity is introduced in two ways. First, the parameter values of the algorithms are varied. Second, different deep reinforcement learning algorithms are used to vary the algorithm type. These algorithms can be adapted to continuous state spaces and some of them even to continuous action spaces which is one major criticism of tabular Q-learning. Additionally, several different economic models are created and used in the experiments broadening the variety of environments in which agents act.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

First, the results of existing studies that collusion between two independent agents may occur in homogeneous settings are confirmed. The level of collusion drops if parameters, algorithm types or both are varied. Specifically, differences between the profit gains of the two agents can be observed for heterogeneous parameters, heterogeneous algorithms and a combination of both. This indicates that firms compete for the best-fitting parameter values and algorithm types. Heterogeneity in parameters and algorithms also leads to lower industry-wide profit gains. In accordance with (Hettich, 2021; Eschenbaum, Mellgren, and Zahn, 2022), the results suggest that it is vital to prevent coordination during the implementation of algorithms in order to prevent collusion.

The structure of this paper is as follows. Section 2 follows with a description of the four economic models that are used in the machine learning experiments. The framework for machine learning, the algorithms and the experimental setup are presented in Section 3. Section 4 contains the results which are discussed in Section 5. Section 6 concludes this paper.

2 Economic models

Five different models are considered as underlying economic environments for the experiments. Next to the logit-demand model used in existing literature, Bertrand, Bertrand-Edgeworth, Cournot and differentiated Bertrand models are used. The environments are restricted to duopoly models with a set of $n = 2$ firms $N = \{1, 2\}$. Firms take simultaneous decisions and - except for the logit-demand model - incur no fixed costs. In the following section, the demand functions as well as the Nash equilibria and collusive outcomes are briefly characterized.

Bertrand

Considering the Bertrand model, the market demand is characterized by

$$D(p) = 1 - p.$$

Firms produce a homogeneous good. Consumers buy from the cheapest seller(s). If the two firms set the same price, any positive demand is split evenly between those firms.

Bertrand-Edgeworth

In addition to the market demand and description of the Bertrand model above, capacity-constrained firms are also considered. Each firm has a capacity of $k_i = 0.6$ with $i \in \{1, 2\}$. As the sum of capacities exceeds the market demand at the competitive price $p = c = 0$, the Nash equilibrium and the collusive solution are the same as in the Bertrand model.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Cournot

The inverse market demand in the Cournot model is assumed to be

$$p(Y) = 1 - Y,$$

where $Y = \sum_{i=1}^{n} y_i$ and firms produce a homogeneous good.

Differentiated Bertrand

The assumption of homogeneous goods is dropped and the market demands in the differentiated Bertrand model are described by

$$D_i(p_i, p_j) = 1 - 2p_i + p_j,$$

where $i, j \in N$ and $j \neq i$.

Logit-demand

Lastly, a logit demand function as in Calvano et al. (2020) is considered:

$$D_i(p_i, p_j) = \frac{\exp\left((2 - p_i)/0.25\right)}{\sum_{m=1}^{2} \exp\left((2 - p_m)/0.25\right) + 1}$$

with $i, j \in N$ and $j \neq i$ and common marginal costs of $c = 1$.

Table 1 gives an overview of the competitive (superscript O) and coordinated (superscript C) prices or quantities and profits for each model per firm $i \in N$. The prices and profits for the logit model in this table are rounded to two decimal places.

Table 1: Competitive and coordinated prices, quantities and profits

Model	p_i^O, y_i^O	π_i^O	p_i^C, y_i^C	π_i^C
Bertrand	0	0	1/2	1/8
Bertrand-Edgeworth	0	0	1/2	1/8
Cournot	1/3	1/9	1/4	1/8
Differentiated Bertrand	1/3	2/9	1/2	1/4
Logit-demand	1.47	0.22	1.93	0.34

3 Experiment design

This section is started with a description of the reinforcement learning framework. The economic models in the previous section are used as environments in which the so-called agents will act. The algorithms are then outlined. The section is finished with an explanation of the experimental setup.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Reinforcement learning environment

The finite Markov decision process is used as a framework for reinforcement learning. At each time step t, the agent observes a state of the environment $S_t \in \mathcal{S}$. After observing this state, the agent takes an action $A_t \in \mathcal{A}$.[1] In time step $t+1$, the agent receives a reward $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$ and observes the next state S_{t+1} (Sutton and Barto, 1998, p. 48). The transition between states is described by the dynamics of the Markov decision process

$$p(s', r|s, a) = \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\},$$

for all $s', s \in \mathcal{S}, r \in \mathcal{R}, a \in \mathcal{A}$. In such an environment, the objective of an agent is to maximize the expected discounted return

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1},$$

where γ is the discount rate (ibid., p. 55). What action an agent selects at time step t is described by a policy $\pi(a|s)$ which gives the probability that $A_t = a$ if $S_t = s$. The state-value function estimates the expected future rewards and is formulated as

$$v_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s \right], \text{ for all } s \in \mathcal{S}.$$

The value of an action a can be stated accordingly by the action-value function for policy π as (ibid., p. 58)

$$Q_\pi(s, a) = \mathbb{E} \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a \right].$$

A policy π_* is called optimal if it yields higher or equally high expected returns than any other strategy (ibid., p. 63).

Algorithms

Two different deep reinforcement learning algorithms are considered in the experiments, DQNs and proximal policy optimization (PPO). DQNs are based on Q-learning where the action-value function, which is based on the Bellman equation, is estimated according to

$$Q_{i+1}(s, a) = \mathbb{E}[r + \gamma \max_{a;} Q_i(s', a')|s, a]$$

where i is the training iteration. Instead of estimating the action-value for each sequence, as would be the case for tabular Q-learning, a function approximator

[1] It is assumed that action sets are independent of the current state, that is the possible actions are the same in each period.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

in the form of an artificial neural network (ANN) is used to estimate these values in DQN algorithms (Mnih et al., 2013, pp. 2–3). Such an ANN consists of units that are interconnected and resemble neurons of nervous systems. Figure 1 shows a graphical representation of an ANN.[2] In this graph, the nodes represent artificial neurons which are called threshold units. These threshold units are connected to each other such that a network is created (Krogh, 2008).

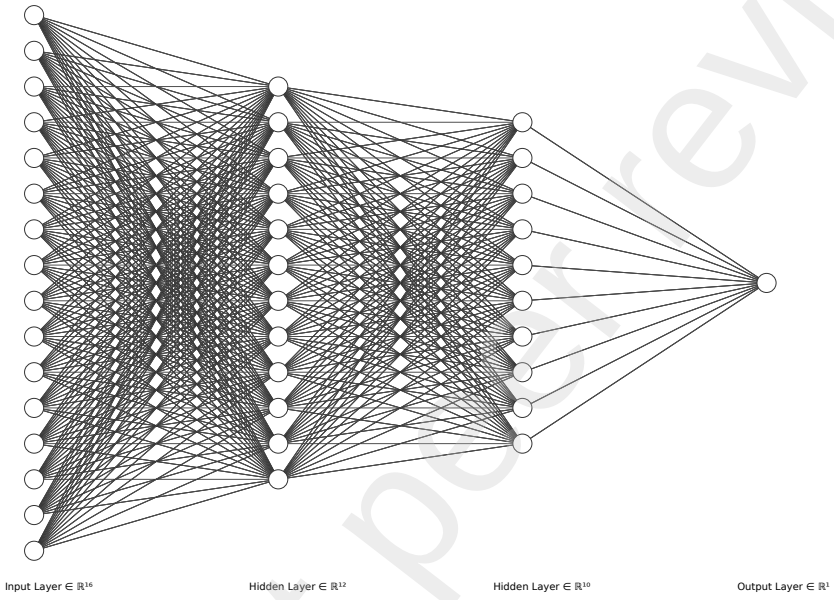

Figure 1: Graphical representation of an artificial neural network.

The processing of signals is influenced by the weights of these connections. ANNs have input layers, output layers and potentially several layers in between, so-called *hidden layers* (Sutton and Barto, 1998, pp. 223–228). The weights are represented by θ. Training takes the form of minimizing the loss function

$$L_i(\theta_i) = \mathbb{E}_{s,a\sim\rho(\cdot)}\left[(y_i - q(s,a;\theta_i))^2\right].$$

In this equation, $y_i = \mathbb{E}_{s'\sim\mathcal{E}}[r + \gamma \max_{a'} q(s',a';\theta_{i-1})|s,a]$ is the target and $\rho(s,a)$ represents a probability distribution over states and actions (addressed in more detail later)[3]. The loss function is minimized with respect to the weights θ_i. The optimization of the loss function is done by stochastic gradient descent (Mnih et al., 2013, p. 3; Sutton and Barto, 1998, pp. 200–204).

Proximal policy optimization is based on the policy gradient method and trust region policy optimization. It combines the advantages of both methods and is therefore scalable, data efficient and robust (Schulman et al., 2017, p. 1).

[2]The graphic has been created using the web tool made by Alex LeNail available at `http://alexlenail.me/NN-SVG/index.html`.

[3]In Mnih et al. (2013), $\mathcal{E}$ represents the environment.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Policy gradient methods with function approximations were proposed by Sutton et al. (1999). This method makes use of a parameterized policy. Such a policy is independent of a value- or action-value function. The policy must, however, be differentiable with respect to its weights. ANNs can be used to compute the action preferences. The method maximizes a performance measure that is based on the policy parameters using stochastic gradient ascent (Sutton and Barto, 1998, pp. 321–326).

Trust region policy optimization is based on the natural policy gradient method. The local approximation of the expected reward is maximized under a certain constraint. This constraint limits the Kullback-Leibler divergence (which measures how a probability distribution is different from another one) for the old policy and the new policy (Schulman et al., 2015). This results in more conservative policy updates and more stability during the training period.

Combining these two methods, PPO maximizes the objective

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]$$

where $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{old}}(a_t|s_t)$ and $\hat{A}_t$ is an estimator of the advantage function (which measures how good an action is compared to the average) at timestep t. The clipping constrains the size of policy updates (Schulman et al., 2017, pp. 2–5).

Experimental setup

The basic experimental design is the same for each economic environment. The baseline results are established for each algorithm by running two independent instances in the respective economic environment. Three variations of the baseline experiments are considered in a second step. First, the parameters of the learning algorithms are altered for one agent to simulate heterogeneous agents which still use the same algorithm type. Second, the type of algorithm is altered for one agent while leaving all parameters at the default values. Lastly, both variations are combined such that the agents differ in the algorithm type and use altered parameter settings.

Agents play against each other for 10,000 timesteps in each experiment. Both algorithms use the same behavior distribution, namely stochastic sampling, to select an action. For stochastic sampling, an action distribution is created from the neural network output. An action is then sampled from this distribution. The training batches which are used in the learning processes consist of 100 samples. Among other settings, the default parameters consist of the discount factor $\gamma = 0.99$, the learning rate for the PPO algorithm $\alpha = 0.00005$ and the learning rate for the DQN algorithm $\alpha = 0.0001$. The action space has $|A| = 10$ elements which an agent can choose. The state spaces are stored in a collection of key-value pairs. The keys are the agent identification strings and the values are the action values. Though both algorithms can be adapted to continuous state spaces and some even to continuous action spaces, discrete state and action spaces are used during all experiments for simplicity.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

The altered parameter settings consist of the discount factor $\gamma = 0.9$, a learning rate of $\alpha = 0.0001$ for each algorithm, and an action set with $|A|=$ 6 elements. The state spaces are adapted accordingly. Similarly to existing studies, the action sets are created by using the Nash equilibrium and collusive outcomes as limits for an interval. Then, for the lower limit, 10 % is subtracted and for the upper limit, 10 % is added. The limits are then used to divide this interval of prices or quantities into equal parts. The central variable of interest is the measure of supra-competitive profits $\Delta_i = \left(\pi_i - \pi_i^N\right) / \left(\pi_i^C - \pi_i^N\right)$ for each firm i which is based on the measure used in Calvano et al. (2020).

Lastly, it should be noted that the aim of this study is not to find the highest possible degree of coordination between agents for each experiment setting. Rather, the fully homogeneous experiments in each environment are used as baselines to which experiments with heterogeneity are compared to. The degree of collusion may differ across environments but is, thus, comparable over experiments in the same environment.

Two tests are used to identify differences in the profit gains. The first test is a Mann-Whitney U test. The null hypothesis of this test is that the distributions are equal and the alternative hypothesis is that the distributions are not equal. Additionally, a Welch's t-test is conducted with a sample size of $n = 100$ to assess differences in the mean. The null hypothesis is that the means are equal. The alternative is that they are not equal.

The differences between environments must be interpreted very carefully. The default and altered parameters are the same for each environment. These values, however, have not been optimized in a grid search which would increase the profit of an agent for a given environment. This was done on purpose to ensure consistency in experiments. The potential to improve parameter values is probably not the same for each environment.

Experiments are carried out using the experiment execution library tune together with the reinforcement learning library RLlib which are both part of the compute framework ray (Liaw et al., 2018; Liang et al., 2018). The tensorflow library for machine learning and artificial intelligence is used (Abadi et al., 2016). The experiments have been customized such that two agents learn and act independently in an environment. The environments have been created according to the above-mentioned economic models.

4 Results

In this section, the experimental results are presented. To explore these results, summary statistics and correlations are reported complemented by a histogram of average profit gains. The focus is then put on how the different settings affect per-firm and industry-wide average profit gains.

Table 2 depicts the mean, the standard deviation and the quartiles together with the minima and maxima of the average profit gains for each of the two agents and for the industry-wide profit gain (average of both agents) over all experiments. The average profit gains of the agents show differences. The mean

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 2: Summary statistics

	$\bar{\Delta}_1$	$\bar{\Delta}_2$	$\bar{\Delta}$
mean	0.5084	0.4902	0.4993
std	0.1445	0.1435	0.1245
min	0.0850	0.0900	0.2349
25%	0.4192	0.3971	0.4152
50%	0.5072	0.4862	0.5052
75%	0.5872	0.5610	0.5726
max	1.0813	1.0294	0.7395

and all quartiles are lower for the second agent. As the second agent sometimes has altered parameter settings, this suggests that these altered parameter settings lead to decreased profit gains. In addition, Figure 2 shows the histogram and quartiles of the industry-wide average profit gain.

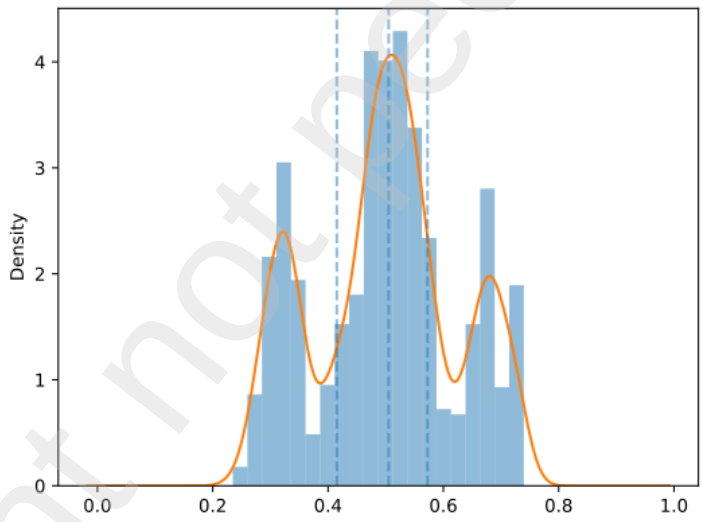

Figure 2: Histogram and quartiles of the industry-wide average profit gain.

Pearson correlation coefficients are reported in Table 3. Consider first the correlation between two measurements of the same agent. All correlation coefficients have the same sign and differ only slightly in size if the two agents are compared. The correlation pairs of different agents show an interesting pattern. A positive correlation can be observed between the pair of minimum and average profit gains, the two minimum profit gains and the two maximum profit gains.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 3: Pearson correlation coefficients

	$\bar{\Delta}_1$	$\Delta_1^{\min}$	$\Delta_1^{\max}$	$\bar{\Delta}_2$	$\Delta_2^{\min}$	$\Delta_2^{\max}$	$\bar{\Delta}$
$\bar{\Delta}_1$	1.0000	0.6110	-0.1853	0.4958	0.6100	-0.2126	0.8658
$\Delta_1^{\min}$	0.6110	1.0000	-0.6063	0.5836	0.9998	-0.5966	0.6907
$\Delta_1^{\max}$	-0.1853	-0.6063	1.0000	-0.1996	-0.6071	0.9633	-0.2225
$\bar{\Delta}_2$	0.4958	0.5836	-0.1996	1.0000	0.5843	-0.1519	0.8638
$\Delta_2^{\min}$	0.6100	0.9998	-0.6071	0.5843	1.0000	-0.5968	0.6905
$\Delta_2^{\max}$	-0.2126	-0.5966	0.9633	-0.1519	-0.5968	1.0000	-0.2109
$\bar{\Delta}$	0.8658	0.6907	-0.2225	0.8638	0.6905	-0.2109	1.0000

Correlation coefficients involving the maximum profit gains and the minimum and average profit gains are negative. The strongest non-trivial correlation can be observed between the minimum profit gains.

Table 4 shows the average profit gains for each agent in the logit environment. The results for all other environments are reported in Section 7 and will be described hereafter. Values are reported for each configuration of algorithms and parameters. Consider first the fully homogeneous settings with PPO agents. The null hypothesis cannot be rejected for either test suggesting no difference between the profit gains of the two agents in each environment. There are low p-values for the Mann-Whitney U test for the Cournot and differentiated Bertrand environments. However, this indicates only a difference in the distribution not necessarily in the means of the samples. For the DQN agents, all tests suggest that there are no differences in the distributions and the means.

Differences between the profit gains of agents can already be observed by introducing heterogeneous parameters. The picture is very clear for DQN. The agent with altered parameter settings has a lower average profit gain compared to its rival in all environments. All tests indicate a significant difference in distributions and means. With PPO agents, the null hypothesis of both tests must be rejected in all except the Bertrand-Edgeworth environment where there is no significant difference between the means. However, it is not very clear how the altered parameter settings affect the profit gains for this type of algorithm. While most agents with the altered parameters have a lower profit gain compared to their rivals, the agent with the altered parameters has a much higher profit gain compared to its rival in the Bertrand environment.

Comparing agents that use two different types of algorithms with the default parameters also shows significant differences in the average profit gains. The null hypothesis of all tests cannot be rejected for experiments with heterogeneous algorithms. Strikingly, the PPO agent always has a higher average profit gain compared to its DQN rival. The difference is robust to the ordering of the agents.

Lastly, the differences between the average profit gains in fully heterogeneous settings are investigated. As expected, the combination of heterogeneous parameters and heterogeneous algorithms also leads to significant differences

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

between the average profit gains of agents. All tests reject the null hypothesis for each environment. As in the experiments with heterogeneous algorithms, the PPO agents have a higher average profit gain in each environment compared to the DQN agent. The effect of the parameter altering is the same for both algorithm types. It reduces the average profit gain of the respective agent. As a result, the difference between the profit gains *increases* if the first agent uses PPO and the second one uses DQN with altered parameters. On the other hand, the difference between the profit gains *decreases* if the first agent uses DQN and the second one uses PPO with altered parameters.

In summary, heterogeneity in parameters and algorithms separately is enough to observe significant differences in the average profit gains of agents. While the effect of heterogeneous parameters is not entirely clear, heterogeneous algorithms clearly show a difference in the average profit gains of agents. The PPO algorithm is more fitting for the environments presented here. Stated otherwise, the PPO agents learn to exploit the DQN agent to some extent. This suggests that firms compete via parameters and algorithm types.

It has been shown that heterogeneity in parameters and algorithm types leads to a change in the average profit gains of agents. In most cases, the profit gains decrease with more heterogeneous settings and increase in only some of the cases. To assess the effect of heterogeneous settings on the whole market, the analysis is extended to the industry-wide profit gain.

Table 5 shows the industry-wide profit gain for the logit environment, and each algorithm type and parameter configuration. Again, the results for all other environments are reported in Section 7 and are discussed hereafter. In these tables, the previously mentioned tests are used to identify differences in heterogeneous settings compared to homogeneous settings which are used as baseline experiments.

As can easily be seen, both tests suggest significant differences in the distribution and means of almost all experiments with heterogeneity compared to the respective baseline experiments. Exceptions are the Cournot and Logit environments. In both of these environments, the industry-wide profit gain is not significantly different from that in the baseline experiment.

Two features stand out with regard to the changes in the industry-wide profit gains. First, profit gains are lower in markets with heterogeneity in each but one environment. For the Cournot experiments with heterogeneous algorithms the profit gains *increase* compared to the homogeneous baselines suggesting that for this environment the two different algorithms complement each other. The second aspect to notice is that (for all significant decreases) heterogeneous algorithms reduce the profit gains further than heterogeneous parameters. The combination of both sources of heterogeneity leads to the highest reductions in profit gains.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 4: Mean profit gains for the logit environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}_1$	$\bar{\Delta}_2$	p (U test)	p (t test)
PPO	hom.	hom.	0.5374	0.5341	.111	.24
PPO	hom.	het.	0.5602	0.5082	$<.001$	$<.001$
PPO	het.	hom.	0.3550	0.3061	$<.001$	$<.001$
PPO	het.	het.	0.3449	0.2553	$<.001$	$<.001$
DQN	hom.	hom.	0.5124	0.5054	.27	.50
DQN	hom.	het.	0.5003	0.4700	.013	.021
DQN	het.	hom.	0.3063	0.3540	$<.001$	$<.001$
DQN	het.	het.	0.2744	0.3301	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 5: Market-wide mean profit gains for the logit environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}$	p (U test)	p (t test)
PPO	hom.	hom.	0.5357	baseline	
PPO	hom.	het.	0.5342	.144	.24
PPO	het.	hom.	0.3305	$<.001$	$<.001$
PPO	het.	het.	0.3001	$<.001$	$<.001$
DQN	hom.	hom.	0.5089	baseline	
DQN	hom.	het.	0.4851	$<.001$	$<.001$
DQN	het.	hom.	0.3301	$<.001$	$<.001$
DQN	het.	het.	0.3023	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

5 Discussion

The present results can be reviewed in the context of related studies. Hettich (2021, p. 10) carried out experiments with heterogeneous algorithms. He used variants of Q-learning and reports an average profit gain of 63 percent for a duopoly of DQN agents. This value is indeed higher than the 54 and 51 percent reported for Logit environments with two PPO and DQN agents respectively. This difference may be attributed to differences in parameters, especially the learning rate which is one of the most important factors. Hettich (ibid., p. 10) uses a learning rate of $\alpha = 0.001$ whereas the default learning rate in this study is $\alpha = 0.00005$ for PPO and $\alpha = 0.0001$ for DQN.

In spite of these differences, the implications of both studies are in line with each other. Hettich (ibid., p. 12) analyses the effect of introducing more firms to the market. He finds that the profit gain decreases to 30 percent for three firms and approaches zero when adding more agents. Additionally and similarly to this study, heterogeneous algorithms – in the form of a tabular Q-learning and a DQN agent – are compared (ibid., p. 14). The main result is that the DQN agent learns to exploit the tabular Q-learning agent and therefore has a much higher profit gain. The same tendency has been found in the experiments of the present study.

Similarly, heterogeneity is also an aspect discussed by Eschenbaum, Mellgren, and Zahn (2022). The authors show that algorithms that were trained in a training environment fail to transfer collusive behavior to slightly changed environments and suggest that this is due to the overfitting of the policies to the policies of rival firms. This overfitting is seen to be caused by large strategy spaces. Restricting the observation space of an algorithm to its own actions reestablishes collusive behavior and enables the transfer of such behavior to new environments. Eschenbaum, Mellgren, and Zahn (ibid., pp. 22–23) conclude that the most important factor to algorithmic collusion is coordination on algorithm design, especially on the parameterization of the environment and the observation space. Though closely related to the present study, the environment and observation space are distinct sources of heterogeneity in addition to those investigated in the present study. This shows that there are many sources of heterogeneity that have an influence on algorithmic collusion.

The previous points stress the conjecture that a high level of collusion is not very likely if firms implement price or quantity-setting deep reinforcement learning algorithms in isolation and firms compete in parameters and algorithm types. This is also supported by Deng (2018, p. 90) who writes that "there are good reasons to suspect that the AI algorithms designed under symmetry do not necessarily fare well in more realistic situations". Coordination on the details of the implementation is necessary to achieve supra-competitive profits. Similarly to Harrington Jr (2020, p. 16), the adoption of highly similar or identical implementations of algorithms results in higher levels of collusion.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

6 Conclusion

This study investigates how algorithmic heterogeneity influences the level of collusion in markets with deep reinforcement learning agents. Heterogeneity is introduced by varying algorithm parameters and algorithm types. The homogeneous baseline experiments confirm the findings in existing studies that supra-competitive outcomes can be observed. Introducing heterogeneity lowers the level of collusion in terms of the average profit gains. Both sources of heterogeneity decrease the level of collusion on their own and also in conjunction.

Heterogeneity can, therefore, be identified as a factor that limits collusion. Firms have an interest in finding optimal parameter values and algorithm types simply to maximize their profit on an individual level. But heterogeneity will decrease if firms use, by chance or by active coordination, very similar or the same parameters or algorithms. This would result in increased prices and profits.

Coordination during the implementation of course could be seen as a means to achieve a supracompetitive outcome that violates antitrust law. Consequently, it seems to be crucial for antitrust authorities to investigate and regulate the implementation processes. This also applies to third-party pricing firms.

In this paper, only a small fraction of a much more complex relationship is analyzed. Several other questions arise from here. The determinants of the size of the differences in the average profit gains would of course be very important to study. The complexity of experiments can be increased of course even further by introducing more firms and changing the time structure of the game. Additionally, there are many other algorithm types. But it could also be very interesting to study the interrelation between human decision-makers and learning algorithms in a market. Only a comprehensive understanding of these issues can ensure that the usage of algorithms in economic contexts is treated properly.

References

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng (2016). "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems". In: arXiv: 1603.04467. URL: http://arxiv.org/abs/1603.04467.

Calvano, Emilio, Giacomo Calzolari, Vincenzo Denicolò, and Sergio Pastorello (2020). "Artificial intelligence, algorithmic pricing, and collusion". In: *American Economic Review* 110.10, pp. 3267–3329.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Deng, Ai (2018). "What Do We Know About Algorithmic Tacit Collusion?" In: *SSRN Electronic Journal* 33.1. ISSN: 1556-5068. DOI: 10.2139/ssrn.3171315. URL: https://www.ssrn.com/abstract=3171315.

Dorner, Florian E. (2020). "Algorithmic collusion: A critical review". In: *SSRN Electronic Journal* 100.2015, pp. 31–35. ISSN: 1556-5068. arXiv: 2004.03107. URL: https://www.ssrn.com/abstract=3548336.

Eschenbaum, Nicolas, Filip Mellgren, and Philipp Zahn (2022). "Robust Algorithmic Collusion". In: December, pp. 1–38. arXiv: 2201.00345. URL: https://arxiv.org/abs/2201.00345v2.

Ezrachi, Ariel and Maurice E. Stucke (2017). *Two Artificial Neural Networks Meet in an Online Hub and Change the Future (Of Competition, Market Dynamics and Society)*. July 2017. ISBN: 9780674545472. DOI: 10.2139/ssrn.2949434.

Gal, Michal S (2019). "Algorithms As Illegal Agreements." In: *Berkeley Technology Law Journal* 34.1, pp. 67–118. ISSN: 10863818. URL: http://10.0.61.163/Z38VM42X86{\%}0Ahttp://search.ebscohost.com/login.aspx?direct=true{\&}db=buh{\&}AN=136573348{\&}site=ehost-live.

Harrington, Joseph E. (2018). "Developing competition law for collusion by autonomous artificial agents". In: *Journal of Competition Law and Economics* 14.3, pp. 331–363. ISSN: 17446422. DOI: 10.1093/joclec/nhy016.

Harrington Jr, Joseph E. (2020). "Third Party Pricing Algorithms and the Intensity of Competition". In: *SSRN Electronic Journal*, pp. 1–28. ISSN: 1556-5068. DOI: 10.2139/ssrn.3723997. URL: https://www.ssrn.com/abstract=3723997.

Hettich, Matthias (2021). "Algorithmic Collusion: Insights from Deep Learning". In: *SSRN Electronic Journal*, pp. 1–19. ISSN: 1556-5068. DOI: 10.2139/ssrn.3785966. URL: https://www.ssrn.com/abstract=3785966.

Klein, Timo (2021). "Autonomous algorithmic collusion: Q-learning under sequential pricing". In: *The RAND Journal of Economics* 52.3, pp. 538–558. ISSN: 0741-6261. DOI: 10.1111/1756-2171.12383. URL: https://onlinelibrary.wiley.com/doi/10.1111/1756-2171.12383.

Krogh, Anders (2008). "What are artificial neural networks?" In: *Nature Biotechnology* 26.2, pp. 195–197. ISSN: 1087-0156. DOI: 10.1038/nbt1386. URL: http://www.nature.com/articles/nbt1386.

Liang, Eric, Richard Liaw, Philipp Moritz, Robert Nishihara, Roy Fox, Ken Goldberg, Joseph E. Gonzalez, Michael I. Jordan, and Ion Stoica (2018). "RLlib: Abstractions for Distributed Reinforcement Learning". In: *35th International Conference on Machine Learning, ICML 2018* 7, pp. 4768–4780. arXiv: 1712.09381. URL: http://arxiv.org/abs/1712.09381.

Liaw, Richard, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E. Gonzalez, and Ion Stoica (2018). "Tune: A Research Platform for Distributed Model Selection and Training". In: 2012. arXiv: 1807.05118. URL: http://arxiv.org/abs/1807.05118.

Maskin, Eric and Jean Tirole (1988). "A Theory of Dynamic Oligopoly, II: Price Competition, Kinked Demand Curves, and Edgeworth Cycles". In: *Econo-*

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

metrica 56.3, p. 571. ISSN: 00129682. DOI: 10.2307/1911701. URL: https://www.jstor.org/stable/1911701?origin=crossref.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller (2013). "Playing Atari with Deep Reinforcement Learning". In: pp. 1–9. arXiv: 1312.5602. URL: http://arxiv.org/abs/1312.5602.

Schulman, John, Sergey Levine, Philipp Moritz, Michael Jordan, and Pieter Abbeel (2015). "Trust region policy optimization". In: *32nd International Conference on Machine Learning, ICML 2015* 3, pp. 1889–1897. arXiv: 1502.05477.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov (2017). "Proximal Policy Optimization Algorithms". In: pp. 1–12. arXiv: 1707.06347. URL: http://arxiv.org/abs/1707.06347.

Schwalbe, Ulrich (2018). "Algorithms, Machine Learning, and Collusion". In: *Journal of Competition Law & Economics* 14.4, pp. 568–607. ISSN: 1744-6414. DOI: 10.1093/joclec/nhz004.

Sutton, Richard S and Andrew G Barto (1998). *SutSutton, R. S., & Barto, A. G. (1998). SuttonBartoSecondBook. In The Lancet (Vol. 258). https://doi.org/10.1016/S0140-6736(51)92942-XtonBartoSecondBook*. Vol. 258. 6685, pp. 675–676. ISBN: 0262193981. arXiv: 1603.02199.

Sutton, Richard S., David McAllester, Satinder Singh, and Yishay Mansour (1999). "Policy Gradient Methods for Reinforcement Learning with Function Approximation". In: *Advances in neural information processing systems* 12. ISSN: 00334081.

7 Appendix

Tables 6 to 9 show the mean profit gains and the p-values of the Mann-Whitney U test and the Welch's t-test for each agent in the Bertrand, Bertrand-Edgeworth, Cournot and differentiated Bertrand environments, respectively. Tables 10 to 13 show the market-wide profit gain and the p-values of the Mann-Whitney U test and the Welch's t-test in the Bertrand, Bertrand-Edgeworth, Cournot and differentiated Bertrand environments, respectively.

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 6: Mean profit gains for the Bertrand environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}_1$	$\bar{\Delta}_2$	p (U test)	p (t test)
PPO	hom.	hom.	0.7251	0.7216	.28	.59
PPO	hom.	het.	0.5969	0.7738	$<.001$	$<.001$
PPO	het.	hom.	0.5283	0.4852	$<.001$	$<.001$
PPO	het.	het.	0.5451	0.4400	$<.001$	$<.001$
DQN	hom.	hom.	0.6759	0.6667	.33	.67
DQN	hom.	het.	0.6938	0.5955	$<.001$	$<.001$
DQN	het.	hom.	0.4855	0.5266	$<.001$	$<.001$
DQN	het.	het.	0.4812	0.5025	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 7: Mean profit gains for the Bertrand-Edgeworth environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}_1$	$\bar{\Delta}_2$	p (U test)	p (t test)
PPO	hom.	hom.	0.7241	0.7207	.223	.51
PPO	hom.	het.	0.6891	0.6840	.042	.30
PPO	het.	hom.	0.4980	0.4554	$<.001$	$<.001$
PPO	het.	het.	0.5229	0.4101	$<.001$	$<.001$
DQN	hom.	hom.	0.6744	0.6651	.26	.59
DQN	hom.	het.	0.7363	0.5519	$<.001$	$<.001$
DQN	het.	hom.	0.4560	0.4966	$<.001$	$<.001$
DQN	het.	het.	0.4623	0.4714	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 8: Mean profit gains for the Cournot environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}_1$	$\bar{\Delta}_2$	p (U test)	p (t test)
PPO	hom.	hom.	0.4236	0.4301	.015	.085
PPO	hom.	het.	0.4432	0.4121	$<.001$	$<.001$
PPO	het.	hom.	0.5501	0.5320	$<.001$	.002
PPO	het.	het.	0.5636	0.5008	$<.001$	$<.001$
DQN	hom.	hom.	0.4028	0.4015	.48	.87
DQN	hom.	het.	0.4021	0.3744	.013	.018
DQN	het.	hom.	0.5359	0.5513	$<.001$	.004
DQN	het.	het.	0.5147	0.5390	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 9: Mean profit gains for the differentiated Bertrand environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}_1$	$\bar{\Delta}_2$	p (U test)	p (t test)
PPO	hom.	hom.	0.5849	0.5777	.026	.069
PPO	hom.	het.	0.6034	0.5309	$<$.001	$<$.001
PPO	het.	hom.	0.3719	0.2960	$<$.001	$<$.001
PPO	het.	het.	0.3642	0.2270	$<$.001	$<$.001
DQN	hom.	hom.	0.5774	0.5679	.1876	.48
DQN	hom.	het.	0.5712	0.5257	.003	.006
DQN	het.	hom.	0.2973	0.3697	$<$.001	$<$.001
DQN	het.	het.	0.2423	0.3428	$<$.001	$<$.001

Preprint not peer reviewed

Table 10: Market-wide mean profit gains for the Bertrand environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}$	p (U test)	p (t test)
PPO	hom.	hom.	0.723343	baseline	
PPO	hom.	het.	0.685346	$<.001$	$<.001$
PPO	het.	hom.	0.506722	$<.001$	$<.001$
PPO	het.	het.	0.492551	$<.001$	$<.001$
DQN	hom.	hom.	0.671305	baseline	
DQN	hom.	het.	0.644631	$<.001$	$<.001$
DQN	het.	hom.	0.506056	$<.001$	$<.001$
DQN	het.	het.	0.491871	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 11: Market-wide mean profit gains for the Bertrand-Edgeworth environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}$	p (U test)	p (t test)
PPO	hom.	hom.	0.722413	baseline	
PPO	hom.	het.	0.686557	$<.001$	$<.001$
PPO	het.	hom.	0.476692	$<.001$	$<.001$
PPO	het.	het.	0.466521	$<.001$	$<.001$
DQN	hom.	hom.	0.669762	baseline	
DQN	hom.	het.	0.644085	$<.001$	$<.001$
DQN	het.	hom.	0.476259	$<.001$	$<.001$
DQN	het.	het.	0.466875	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 12: Market-wide mean profit gains for the Cournot environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}$	p (U test)	p (t test)
PPO	hom.	hom.	0.426875	baseline	
PPO	hom.	het.	0.427647	.37	.77
PPO	het.	hom.	0.541067	$<.001$	$<.001$
PPO	het.	het.	0.532219	$<.001$	$<.001$
DQN	hom.	hom.	0.402151	baseline	
DQN	hom.	het.	0.388277	.002	.003
DQN	het.	hom.	0.543600	$<.001$	$<.001$
DQN	het.	het.	0.526841	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554

Table 13: Market-wide mean profit gains for the differentiated Bertrand environment

Policy agent 0	Algorithms	Parameters	$\bar{\Delta}$	p (U test)	p (t test)
PPO	hom.	hom.	0.581263	baseline	
PPO	hom.	het.	0.567126	$<.001$	$<.001$
PPO	het.	hom.	0.333966	$<.001$	$<.001$
PPO	het.	het.	0.295595	$<.001$	$<.001$
DQN	hom.	hom.	0.572678	baseline	
DQN	hom.	het.	0.548471	$<.001$	$<.001$
DQN	het.	hom.	0.333493	$<.001$	$<.001$
DQN	het.	het.	0.292547	$<.001$	$<.001$

Preprint not peer reviewed

This preprint research paper has not been peer reviewed. Electronic copy available at: https://ssrn.com/abstract=4578554